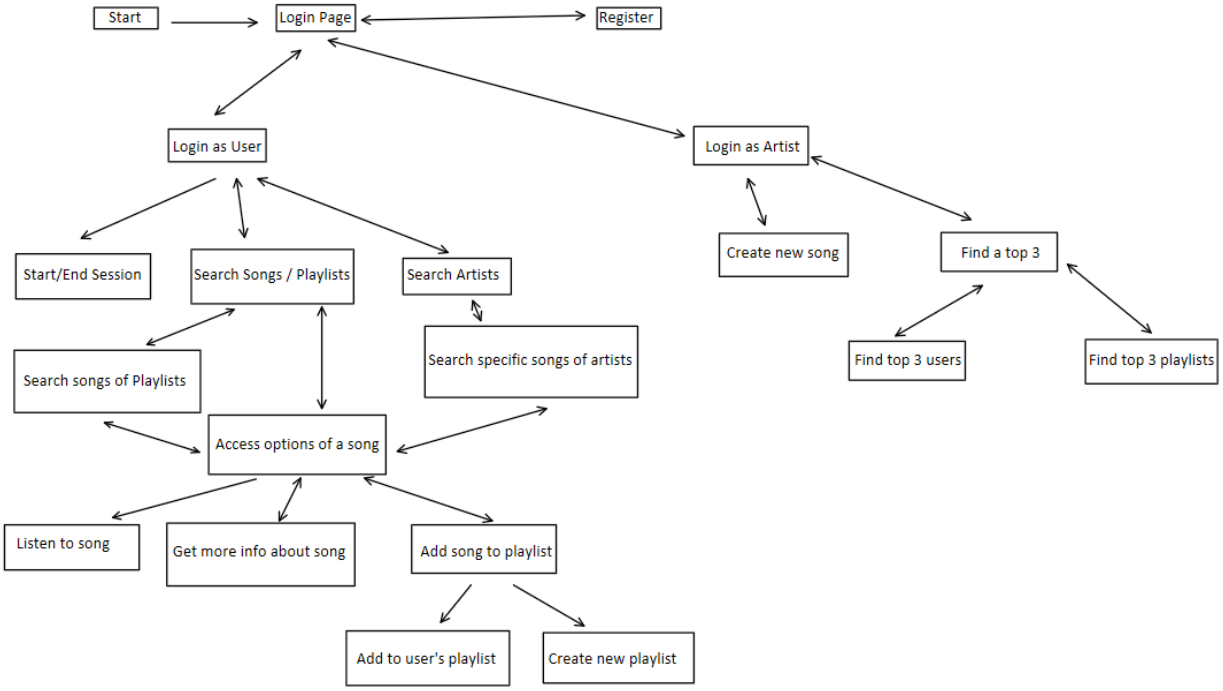
A General Overview Of The System With a Small User Guide

For our program, we use the language python and one of its package Tkinter (with its “tk interface”). With those, we developed different pages that each fulfilled a role and offered multiple functionalities to users and artists. Below is a summarized chart of each pages and steps of the program:



A Detailed Design of The Software - Components Required to Deliver The Major Functions Of The Application

The following functions are used during the login and register parts of the app:

- **home():** Creates the home page and all the buttons and its labels.
- **check():** called when the login button is pressed, verifies that the ID entered matches an ID in the database
- **choose():** called when the id entered matches both a user and an artist, creates à page to make the user choose between the two types of accounts and goes to the password page when the selection is made.
- **Validate(u):** called after the ID has been checked and validated. It verifies that the password entered matches the account.
- **registerPage():** Called when the register button is pressed, creates à page for the user to create à new account as well as all its buttons. Once the user enters the values and clicks the register button, it will verify that all fields are entered, that the ID does not already exist and that the two passwords match. Will display an error message accordingly. If all conditions are met, will bring the user to the user page. Can also return to the home page when the button is pressed.

The following functions are called on the user page interface:

- **StartSession():** It first checks if the user is already in a session, if they are, it display à message telling the user. If the user is not already in a session, it will find à session number that is not already used for the user and create à session with that session number and with à start date of the current time.
- **searchSongsAndPlaylists()**: it is called when the user presses the “search song/playlist” button. This function uses the entered keywords to find all the relevant songs and playlists (using SQL) and passes them to displaySongsPlaylist() (as tuples)
- **displaySongsPlaylist()**: called when the user presses the “search song/playlist” button (after searchSongsAndPlaylists()). This function displays all the songs and playlists that were passed to it and transforms them into buttons (for additional functionalities). If we need to display more than 5 playlists and songs, we can scroll up and down the list to display 5 playlists/songs buttons at a time. There is also a return button that returns the user to the previous page.
- **searchArtists()**: called when the user presses the “search artists” button. This function uses the entered keywords to find all the relevant artists (using SQL) and passes them to displayArtists() (as tuples)
- **displayArtists()**: called when the user presses the “search artists” button (after searchArtists()). This function displays all the artists that were passed to it and transforms them into buttons (for additional functionalities). If we need to display more than 5 artists, we can scroll up and down the list to display 5 artist buttons at a time. There is also a return button that returns the user to the previous page.
- **endSession():** This function is called when the End Session button is clicked on the user page. It first checks if the user is in a session. If they aren’t, it will display a message to tell the user they can’t close à session without being in one. If the user is in à session, it will update the sessions table for this session and set its end date as now.
- **reset():** Called when the Logout or the EXIT button are clicked. it resets all the global variables, notably the id and passwords.

The following functions are called when the buttons for playlists, artists or songs are pressed on**:**

- **determineContent()**: it is called whenever a user clicks on a playlist, artist or song button. This function identifies which category this button is in (playlist, artist or song). If it is a playlist or artist, the function passes the tuple to the songsOfPlaylist() or songsOfArtist() (respectively) to find all the songs related to the playlist or artist. Then, it passes the result to the displaySongs() function, which will display all of the songs in the result. If the tuple was a song (the third case), we simply pass it to the songMenu() function, which will display the options of the song.
- **songsOfPlaylist()**: when called, this function uses the playlist pid passed to it to find all the songs that the playlist contains (using SQL) and returns all of those songs as tuples
- **songsOfArtist():** when called, this function uses the artist aid passed to it to find all the songs that the artist has performed (using SQL) and returns all of those songs as tuples
- **displaySongs():** when called, this function displays all the songs that were passed to it as buttons (for additional functionalities). If we need to display more than 5 songs, we can

scroll up and down the list to display 5 songs buttons at a time. When clicked on it, a song button will call the songMenu() function and pass the song data to it. There is also a return button that returns the user to the previous page.

- **songMenu():** when called, this function displays the three different options for the song that was passed as buttons. Those are 1) the “listen to the song” button, which will call and pass the song data to listenToSong(). 2) The “get more info” button, which will call and pass the song data to infoAboutSong(). 3) The “add to playlist” button, which will call and pass the song data to addSongToPlaylist(). There is also a return button to go back to the previous page.

The following functions are for the functionalities relating to the song:

- **listenToSong()**: when called, the function checks if a listening session has been started. If not, it starts one the continues. It then either creates a new row in the listen table for the song that is being listened to, or increases the “count” field of the corresponding row by 1.
- **infoAboutSong()**: when called, it displays the information about the song that was passed it. There is also a return button to go back to the song functionalities.
- **addSongToPlaylist()**: when called, this function will offer to add the song (that was passed to it) to the user’s playlists or create a new playlist where the user can add the song. A return button also returns to the song functionalities.

The following functions are called on the artist page interface:

- **addnewSong():** This function is called when the New song button is pressed by the artist. It creates a to allow the artist to enter the name and duration of the new song they wish to add, as well at the Add song button. When that button is pressed, it will verify that the data is entered correctly, add the song to the perform and the songs tables and take the artist to the add artists page.
- **addartistPerform():** called after the Add song button is pressed. Creates a page for the artist to add the ID of all other artists that performed on the new song. If that section is left empty, no new entry will be added to the perform table. But if there are entries, it will verify that the IDs match existing artists and will add them to the perform table with the new song.

Our Testing Strategy

To test the login part of the app, we tried using injection to make sure that our code was resistant to it. We also tried entering non existent IDs and incorrect passwords to make sure that it displayed the correct error messages. For the register page, we tried entering IDs that already existed to make sure it wouldn’t allow it, as well as IDs that were longer than the ID length in the database. We also tested what would happen if the fields in all pages were left empty and ensured that an appropriate response was established. To test if the user page was working correctly, we tried to search different songs, playlists and artists (with sometimes incorrect names in the entries) and then tested the functionalities of playlists, songs and artists (by clicking on their respective buttons). To test the artist page, we tried using the find top buttons and manually verified the results with what was in the database. We also tested the add new song button by trying to add a song that already existed or entering non existing artist IDs in the add performers part.